

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平11-272466

(43)公開日 平成11年(1999)10月8日

(51) Int.Cl.⁶

G O 6 F 9/38

識別記号

3 1 0

350

FI

G O 6 F 9/38

310 F

3 5 0 B

350A

審査請求 未請求 請求項の数 7 O L (全 11 頁)

(21)出題番号

特照平11-25857

(22) 出願日

平成11年(1999)2月3日

(31)優先權主張番号 09/021134

(32) 優先日 1998年2月10日

(33)優先權主張国 米国 (US)

(71) 出國人 390009531

インターナショナル・ビジネス・マシーンズ・コーポレーション

INTERNATIONAL BUSINESS
MACHINES CORPORATION

アメリカ合衆国10504、ニューヨーク州
アーモンク (番地なし)

(72)発明者 シーシウン・スティーブン・タン

アメリカ合衆国78726 テキサス州オースチン パープルック・ドライブ 9923

(74)代理人 弁理士 坂口 博 (外1名)

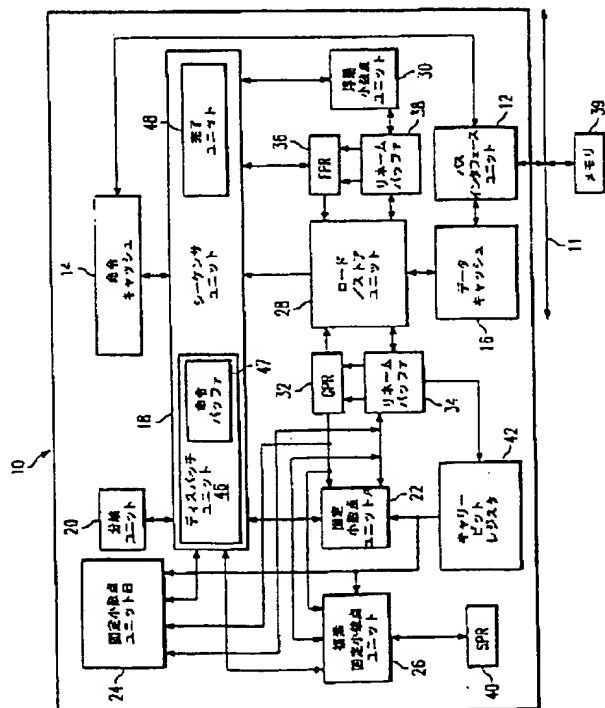
最終頁に続く

(54)【発明の名称】 ロード／ロード検出及びリオーダー方法及び装置

(57) 【要約】

【課題】 本発明は、データ・コヒーレンスを侵害する可能性のあるアウト・オブ・オーダー命令を検出してリオーダーする手段に関する。

【解決手段】 本発明は、各項目がコンピュータ・マイクロプロセッサ内の命令に対応する命令データの項目を保持するミスキュー・テーブルを含む。各項目内の命令データは、i) 命令のアドレス情報と、ii) ミスキュー・テーブル内の他の命令を基準にした命令の順序を示す命令の順序情報と、iii) データが変更された可能性を示す、命令のデータ変更情報と、iv) 前記項目に対応するより古い命令より前により新しい命令が完了したことを示すアウト・オブ・オーダー情報とを含む。本発明は、完了した命令のアドレスをミスキュー内のアドレス情報項目と比較するアウト・オブ・オーダー比較機構も含む。



【特許請求の範囲】

【請求項 1】命令のアウト・オブ・オーダー実行のための装置であって、

a) 各項目がコンピュータ・マイクロプロセッサ内の命令に対応する命令データの項目を保持するミスキュー・テーブルを含み、前記命令データは、

i) 命令のアドレス情報と、

ii) 他の命令を基準にした前記対応する命令の順序を示す命令の順序情報と、

iii) アドレス情報フィールドに対応するアドレスにあるデータを使用するより新しい命令が現行項目より前に完了したことを示すアウト・オブ・オーダー情報と、

iv) 前記アドレス情報フィールドに対応する前記アドレスにあるデータが変更された可能性を示す、命令のデータ変更情報とを含み、

前記装置は、

b) 比較された項目内のアドレス情報フィールドに対応するアドレスにあるデータを使用する、より新しい命令が前記比較された項目より前に完了した場合、前記比較された項目のアウト・オブ・オーダー情報フィールドが設定される、前記ミスキュー・テーブル内の項目に前記アウト・オブ・オーダー情報フィールドを設定するアウト・オブ・オーダー比較機構と、

c) 前記ミスキュー・テーブル内の前記アドレス情報フィールド内のアドレス情報を、変更された可能性のあるアドレスと比較する変更比較機構とを含み、前記変更されたアドレスが比較された前記命令項目内の前記アドレス情報と一致する場合、前記項目内の前記変更フィールドが、前記アドレスにある変更されたデータを示すようにマークされる装置。

【請求項 2】アウト・オブ・オーダー命令と変更されたデータとを示す命令項目に対応する命令のすべての後続命令を取り消す、請求項 1 に記載の装置。

【請求項 3】前記アウト・オブ・オーダー比較機構が項目内のすべてのデータ・アドレス情報を比較して前記アドレスが一致するかどうかを判断する、請求項 1 に記載の装置。

【請求項 4】前記アウト・オブ・オーダー比較機構が、項目内の前記データ・アドレス情報の一部を比較して、前記アドレスが一致するかどうかを判断する、請求項 1 に記載の装置。

【請求項 5】データ・コヒーレンシ侵害を生じさせる可能性のあるマイクロプロセッサ内のアウト・オブ・オーダー命令を検出する方法であって、

a) データ・アドレスを有する前記マイクロプロセッサ上で実行する新規命令を作成するステップと、

b) 前記新規命令が実行されるように設定されている場合、

i) 前記命令のデータ・アドレスをミスキュー・テーブル内の、前の命令に対応し、アドレス情報と命令順序情

報とアウト・オブ・オーダー標識と変更データ標識とを含む既存の命令項目と比較し、

ii) 前記新規命令のデータ・アドレスがミスキュー・テーブル内の項目のアドレス情報と一致する場合、前記ミスキュー・テーブル内の前記一致項目をアウト・オブ・オーダー命令にするステップと、

c) 前記新規命令が実行されるように設定されていない場合、前記ミスキュー・テーブル内の前記新規命令の項目を作成し、前記新規命令のデータ・アドレスが前記新規命令項目のアドレス情報に入れられ、前記命令項目の順序情報に前記命令の順序に関する情報が入れられる方法。

【請求項 6】d) 前記ミスキュー・テーブルを絶えず走査して、実行するように設定されている命令に対応する項目がないか調べるステップと、

e) 前記ミスキュー・テーブル内の項目に対応する命令が実行されるように設定されている場合、

i) 実行するように設定されている前記命令に対応する前記項目のアドレス情報を、実行するように設定されている前記命令より古い命令に対応する前記ミスキュー・テーブル内の項目のアドレス情報と比較するステップと、

ii) 前記ミスキュー・テーブル内に、実行するように設定されている前記命令より古い命令に対応する項目が見つかった場合、及び実行するように設定されている前記命令に対応する前記項目のアドレス情報が前記より古い命令に対応する前記項目内のアドレス情報と一致する場合、前記より古い命令に対応する前記項目をアウト・オブ・オーダーとしてマークするステップと、

iii) 実行するように設定されている前記命令に対応する前記項目を前記ミスキュー・テーブルから除去するステップとをさらに含む、請求項 5 に記載の方法。

【請求項 7】f) データ変更事象がないか絶えず走査するステップと、

g) データ変更事象が発生した場合、

i) 前記変更されたデータのアドレスを前記ミスキュー・テーブルにブロードキャストするステップと、

ii) 前記変更されたデータのアドレスを前記ミスキュー・テーブルの項目内のアドレス情報と比較するステップと、

iii) 前記変更されたデータのアドレスと前記ミスキュー・テーブル内の項目のアドレスが一致する場合、前記一致するアドレスを有する前記ミスキュー・テーブル内の項目を変更されたものとしてマークするステップとをさらに含む、請求項 6 に記載の方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は一般にはコンピュータ・プロセッサの分野に関し、より詳細には、単一マイクロプロセッサ・チップ上に集積されたプロセッサに関

3

する。さらに詳細には、本発明は、特に複数CPUシステムにおける、アウト・オブ・オーダー処理に内在するデータ・コヒーレンシ問題の検出とは正に異なる。

【0002】

【従来の技術】マイクロプロセッサの一層の高速化は、現在のプロセッサ設計の主要な目標の1つである。プロセッサのパフォーマンスを向上させるために多くの異なる技法が採用されてきた。プロセッサ・パフォーマンスを大幅に向上させる1つの技法は、キャッシュ・メモリの使用である。本明細書では、キャッシュ・メモリとは、マイクロプロセッサ自体の上に形成されており、その結果、マイクロプロセッサ・チップから離れた位置にあるRAMや磁気ディスクなどの他のタイプのメモリよりもアクセス時間ははるかに高速な1組の記憶場所を指す。頻繁に使用されるデータのコピーをキャッシュに記憶することによって、プロセッサはそのデータが必要なときにキャッシュにアクセスすることができ、その情報入手するために「オフ・チップ（チップ外）」まで行く必要がなく、その結果、プロセッサのパフォーマンスが大幅に強化される。

【0003】しかし、キャッシュ・メモリには特定の問題が付随する。具体的には、1つのシステムで複数のプロセッサが使用され、同じデータを必要とする場合に大きな問題がある。この場合、システムは要求されるデータが整合するように、すなわちその時点でそのプロセッサにとって有効であるように保証する必要がある。データが1つのプロセッサのキャッシュに記憶されており、別のプロセッサがそれと同じ情報を要求する場合にも問題がある。

【0004】スーパースカラ・プロセッサは命令を非プログラム順序（アウト・オブ・オーダー）で実行することができるため、従来のスカラ・プロセッサより有利なパフォーマンスを実現する。この方法により、実行速度の遅い命令を保留にしている間にプロセッサ上の他の資源を使用して実行することができる後続命令が、実行の遅い方の命令によって停止することがなくなる。

【0005】典型的なアーキテクチャでは、すべての命令が1つのデータを必要とする場合、プロセッサはまずオンボード・キャッシュに行き、そのデータがオンボード・キャッシュに入っていないかどうか調べる。キャッシュによっては、2つの外部ポートを有するものがあり、キャッシュをインタリーブさせることができる。これは、たとえば図1で、キャッシュ100が2つのキャッシュ・バンク140及び130を有することを意味する。一方のキャッシュ・バンクは奇数アドレス用とすることができ、他方のキャッシュ・バンクは偶数アドレス用となる。

【0006】内部的には、各キャッシュ・バンク140及び130キャッシュが内部入力ポート（図示せず）を有し、このポートに対してキャッシュ要求のアドレス情

4

報が入力される。図1で、アドレスA1のデータはキャッシュ・バンク130内のキャッシュ・ライン110に記憶され、アドレスA2のデータはキャッシュ・バンク140内のキャッシュ・ライン120に記憶される。キャッシュ100は入力データ用の2つの外部ポートであるポート180とポート190を有する。

【0007】キャッシュ要求1は、命令1（図示せず）のキャッシュ要求を示し、要求2は命令2（図示せず）のキャッシュ要求を示す。命令1は命令2より古い命令であり、これは命令2よりも前に実行する必要があることを意味する。米国テキサス州オースチンのIBMコーポレーションのPowerPC（TM）プロセッサのようにスーパースカラ・プロセッサが複数のロード・ユニットを有する場合、両方の命令が同時にキャッシュ要求を行うことができる。図の例では、命令2と命令1がアドレスA1にあるデータにアクセスを試みようとして、キャッシュ100にそれを行う要求を渡している。

【0008】バンク130には内部入力ポートが1つしかないため、同時に両方のキャッシュ要求を処理することができない。これは、キャッシュ100のインタリーブ性によるものである。

【0009】図2に、キャッシュ要求2がキャッシュ要求1よりも前にキャッシュ・バンク130にアクセスするとどうなるかを示す。キャッシュ要求2はキャッシュ・バンク130に必要なデータにヒットする。しかし、キャッシュ要求1は少なくとも次のサイクルまでキャッシュ・バンク130にアクセスすることができない。したがって、より新しい命令2は必要なデータを、より古い命令1が入手するよりも前に入手することができる。この場合、このポート割り振り競合のために、この場合、より新しい命令2がより古い命令1より前に完了することができる。

【0010】これと同じ順序づけ問題は、より古い命令がキャッシュでミスし、より新しい命令がヒットした場合にも起こり得る。ミスは、データのアドレスがメモリ管理ユニット内に見つからない場合に起こり、その場合、メモリ管理ユニットはそのデータをより上位のメモリから取り出すように要求しなければならない。ヒットは、データのアドレスとデータの両方に、メモリ管理ユニットとキャッシュを介してアクセスすることができる場合に起こり、そのデータはそれを待っている命令に出力することができる。

【0011】データの真アドレスが2つの異なる有効アドレスによって表されている場合、両方の命令が同じデータにアクセスを試み、より古い命令のキャッシュ・ミスの後により新しい命令によるキャッシュ・ヒットがあるという状況が発生する。より新しい命令によって要求された有効アドレスとそのデータがすでにメモリ管理ユニットとキャッシュによってアクセス可能な場合であって、メモリ管理ユニットとキャッシュでより古い命令の

10

20

30

40

50

アドレスとデータにアクセスすることができない場合にも、より古い命令と同じデータにアクセスするより新しい命令がより古い命令より前に完了することができるという状況が起こる。

【0012】マルチプロセッサ・システムでは、1つのプロセッサでのキャッシュ・ミスによって、システム内の他のプロセッサに対する「スヌープ」要求をトリガすることができる。このスヌープ要求は、他のプロセッサに対して、「スヌープ」されているデータが別のプロセッサによって要求されていることを示し、他のプロセッサは求められているアドレスがそれぞれのプロセッサ自体のキャッシュに入っているかどうかを調べるように指示する。プロセッサのキャッシュに入っている場合、メイン・メモリ・データを整合させる必要がある。すなわち、システム状態の正しい現在の状態を反映するように更新する必要がある。

【0013】スーパースカラ・アーキテクチャに関しては、ロード命令がアウト・オブ・オーダーで終了することがあるため、言い換えると、新しい命令が古い命令よりも前に完了としてマークされることがあるため、この問題はさらに大きくなる。すなわち、新しい命令が古い命令よりも前に実行されるように設定されているものとマークされることがある。したがって、2つのロード命令が同じキャッシュ場所をアドレス指定し、実際には1つのデータが古い命令より前に新しい命令に供給されることがある。したがって、新しい命令は、アウト・オブ・オーダーで完了するマークが付けられ、それによってその命令の完了に不正なデータが使用される可能性がある。後続のロード命令がより古いロード命令を迂回する場合、より古いロード命令は、元のプログラム順序に基づいて受け取るはずだったデータよりも新しいデータを入手する可能性がある。

【0014】このコヒーレンシ問題の従来の解決策としては、ロード・キューにページ索引と実アドレスを、ID及び有効ビットと共に保持する。IDはロード命令のプログラム順序を示す。

【0015】上述の項目に加えて、ロード・キュー項目は、そのアドレスのキャッシュ・ライン項目が変更されたかどうかを示す変更フィールドも保持する。ストア命令やスヌープ要求などのキャッシュ・アクセスによって、キャッシュ・ラインが変更されたことが示されると、ロード・キューが探索される。それと同じラインの項目が含まれている場合、変更の可能性を示すように変更ビットが設定される。

【0016】後続のロードは、ロード・キュー項目の比較を行うことになる。ロード・キュー内で同じラインが保留中であり、変更済みとマークされている場合、IDフィールドを検査する。現在のラインが保留中だったラインよりも古く、変更されている場合、ロード・キュー内の保留ロードが取り消され、その後続ロードの後に再

実行される。これによって、より古いロードがより新しいロードよりも新しいデータを持つという問題が回避される。

【0017】

【発明が解決しようとする課題】

【課題を解決するための手段】本発明は、ロード・キューをなくし、不正なデータを使用して終了した可能性がある命令を取り消すだけの新規な手段を提供する。

【0018】本発明は、拒否されたキャッシュへのアクセスの試行や、命令が完了できないその他の理由を保持するミスキュー・テーブルを提供する。

【0019】好ましい実施形態では、すべての命令がミスキュー・テーブル内に初期項目を作成する。その命令のデータがキャッシュに入っており、入手可能な場合、ミスキュー・テーブル内のその命令項目がミスキュー・テーブルから取り出される。これは、プロセッサが、順序づけ問題の可能性がないかミスキュー・テーブルを探索してから行う。次に、その命令には、プロセッサのシーケンシング・ユニットによって、キャッシュ内で見つかったデータを使用して完了のマークが付けられる。

【0020】古い命令が新しい命令の後に完了し、新しい命令と古い命令が同じアドレスにあるデータにアクセスする場合、順序づけ問題が起こる。

【0021】命令のデータをキャッシュで入手できない場合、あるいは他の何らかの理由で完了できない場合、ミスキュー・テーブル内に作成された命令項目は、命令が完了可能になるまでミスキュー・テーブルにとどまる。キャッシュ内でデータを入手できるようになった場合など、命令が完了可能になると、命令には完了のマークが付けられる。次に、命令項目はミスキュー・テーブルから削除される。

【0022】命令が完了可能になると、それに付随する項目がミスキュー・テーブルから削除される。ミスキュー・テーブルで、同じアドレスを持つ、前の命令項目がないか探索を行う。完了する項目と同じアドレスを持つ前の命令項目が見つかった場合、前の命令項目はまだ完了していないため、アウト・オブ・オーダーとしてマークされる。前述のように、これはたとえば、命令が同じデータ・アドレスに別名を付け、より古い命令にデータが使用可能になっていることがキャッシュから通知されていない場合に起こることがある。

【0023】すべての命令がミスキュー・テーブルに項目を作成して完了可能な命令項目が次のサイクルで引き出されるか、拒否された命令のみがミスキュー・テーブルに項目を作成するかを問わず、機能上の結果は同じであることに留意されたい。この結果とは、ただちに完了する命令に対応する命令項目はミスキュー・テーブルにはとどまらず、ただちには完了しない命令に対応する命令項目のみがミスキュー・テーブル内に残ることである。

【0024】これによって、すべての現行命令が有効であって実行するように設定されていると見なされるか、データを待つミスキュー・テーブルに入れられている対応する項目を有するとみなされるため、ロード・キューを備える必要がなくなる。

【0025】現行命令が完了した場合、データのアドレスをミスキュー・テーブル内の項目と照合して一致があるかどうかを調べる。一致するものがある場合、ミスキュー・テーブル内の一致項目がアウト・オブ・オーダーとしてマークされる。すなわち、アドレスからデータにアクセスする、より新しい命令が、同じアドレスに対する古い命令よりも先に完了するように設定されており、次に、より古い命令がアウト・オブ・オーダーで完了し、ミスキュー・テーブル内のその項目がそれに従ってマークされることになる。

【0026】さらに、キャッシュがミスキュー・テーブルに暫時、項目を有する命令のために有効なデータを返すと、その項目に対応する命令が完了に設定される。そのアドレスにあるデータにアクセスするまだ完了していない命令に対応する古い命令項目についても、ミスキュー・テーブルで同様の探索が行われる。一致する物が見つかったと、一致する古い項目はアウト・オブ・オーダーとしてマークされる。

【0027】スヌープ要求など、データ・コヒーレンシ変更事象が発生した場合、ミスキュー・テーブルに問い合わせる。データ・コヒーレンシ変更事象と同じアドレスを持つ項目には変更済みのマークが付けられる。

【0028】ミスキュー・テーブルから命令項目が解放されると、命令を完了する。プロセッサは次に、特定の事象が発生したかどうかを判断する。ミスキュー・テーブル内の命令項目によって、その項目に対応する命令がアウト・オブ・オーダーであってしかも変更されている場合、そのアウト・オブ・オーダーかつ変更済み命令の後に実行されると思われる命令は取り消され、そして再実行され、したがってデータ・コヒーレンシが保たれる。

【0029】

【発明の実施の形態】図3は、本発明により情報を処理するプロセッサ・システム10を示すブロック図である。好ましい実施形態では、プロセッサ10は米国テキサス州オースチンのIBMコーポレイションのPowerPC™プロセッサなどの単一集積回路スーパースカラ・マイクロプロセッサである。したがって、後で詳述するように、プロセッサ10は様々なユニット、レジスタ、バッファ、メモリ、及びその他の部分を含み、それらはすべて集積回路によって形成されている。また、好ましい実施形態では、プロセッサ10は縮小命令セット・コンピューティング（「RISC」）技法に従って動作する。図3に示すように、システム・バス11がプロセッサ10のバス・インタフェース・ユニット（「BI

U」）12に接続されている。BIU12は、プロセッサ10とシステム・バス11との間の情報の伝送を制御する。

【0030】BIU12は、プロセッサ10の命令キャッシュ14とデータ・キャッシュ16とに接続されている。命令キャッシュ14は、シーケンサ・ユニット18に命令を出力する。シーケンサ・ユニット18は、命令キャッシュ14からのこのような命令に回答して、プロセッサ10の他の実行回路に命令を選択的に出力する。

【0031】好ましい実施形態では、プロセッサ10の実行回路は、ディスパッチ・ユニット46と完了ユニット48の実行ユニットを含むシーケンサ・ユニット18に加えて、複数の実行ユニット、すなわち分岐ユニット20と、固定小数点ユニットA（「FXUA」）22と、固定小数点ユニットB（「FXUB」）24と、複素固定小数点ユニット（「CFXU」）26と、ロード／ストア・ユニット（「LSU」）28と、浮動小数点ユニット（「FPU」）30とを含む。FXUA22、FXUB24、CFXU26、及びLSU28は、それぞれのソース・オペランド情報を汎用アーキテクチャ・レジスタ（「GPR」）32及び固定小数点リネーム・バッファ34から入力する。さらに、FXUA22とFXUB24は、キャリー・ビット（「CA」）レジスタ42から「キャリー・ビット」を入力する。FXUA22、FXUB24、CFXU26、及びLSU28は、それぞれの演算の結果（目的オペランド情報）を出力して、固定小数点リネーム・バッファ34に記憶する。また、CFXU26は、特殊目的レジスタ（「SPR」）40との間でソース・オペランド情報と目的オペランド情報を入出力する。

【0032】FPU30は、そのソース・オペランド情報を浮動小数点アーキテクチャ・レジスタ（「FPR」）36と浮動小数点リネーム・バッファ38から入力する。FPU30は演算の結果（目的オペランド情報）を出力して浮動小数点リネーム・バッファ38内の選択された項目に記憶する。

【0033】シーケンサ・ユニット18は、GPR32及びFPR36との間で情報を入出力する。シーケンサ・ユニット18から、分岐ユニット20が命令と、プロセッサ10の現在の状態を示す信号とを入力する。分岐ユニット20は、このような命令及び信号に回答して、プロセッサ10によって実行される命令のシーケンスを記憶する適切なメモリ・アドレスを示す信号を（シーケンサ・ユニット18に）出力する。シーケンサ・ユニット18は、分岐ユニット20からのこのような信号に回答して、命令キャッシュ14からの指示された命令シーケンスを入力する。命令シーケンスのうちの1つまたは複数の命令が命令キャッシュ14に記憶されていない場合、命令キャッシュ14はシステム・バス11に接続されているシステム・メモリ39から（BIU12とシス

テム・バス11を介して)その命令を入力する。

【0034】シーケンサ・ユニット18は命令キャッシュ14からの命令入力にตอบสนองして、ディスパッチ・ユニット46を介して、実行ユニット20、22、24、26、28、及び30のうちから選択された実行ユニットに選択的にディスパッチする。各実行ユニットは、特定の命令クラスの1つまたは複数の命令を実行する。たとえば、FXUA22とFXUB24はソース・オペランドに対して加算や減算、AND、OR、XORなどの第1のクラスの固定小数点数演算を実行する。CFXU20はソース・オペランドに対して固定小数点乗算や除算など第2のクラスの固定小数点演算を実行する。FPU30は、ソース・オペランドに対して浮動小数点乗算や除算などの浮動小数点演算を実行する。

【0035】プロセッサ10は、実行ユニット20、22、24、26、28、及び30のうちの様々な実行ユニットで同時に複数の命令を処理することによって高パフォーマンスを実現する。したがって、各命令はいくつかの段階から成るシーケンスとして処理され、各段階は他の命令の段階と並列して実行可能である。このような技術は「パイプライン処理」と呼ばれる。好ましい実施形態の重要な態様では、命令は通常6段階、すなわち、フェッチ、デコード、ディスパッチ、実行、完了、及びライトバックの各段階で処理される。

【0036】好ましい実施形態では、各命令は命令処理の各段階を完了するのに1マシン・サイクルを必要とする。それにもかかわらず、ある種の命令(たとえばCFXU26によって実行される複素固定小数点命令)は、複数のサイクルを必要とすることがある。したがって、特定の命令の実行と段階と完了段階との間に、先行命令の完了に要する時間の変動に応じて変動する遅延が生じることがある。

【0037】LSU28は、ロード命令にตอบสนองして、データ・キャッシュ16から情報を入力し、その情報をリネーム・バッファ34及び38のうちの選択されたリネーム・バッファにコピーする。その情報がデータ・キャッシュ16に記憶されていない場合、データ・キャッシュ16はその情報をシステム・バス11に接続されているシステム・メモリ39から(BIU12とシステム・バス11を介して)入力する。さらに、データ・キャッシュ16は、データ・キャッシュ16からシステム・バス11に接続されているシステム・メモリ39に(BIU12とシステム・バス11)を介して情報を出力することができる。

【0038】図4を参照すると、本発明の一実施形態によりロード命令などの命令を処理する回路を図示する略図が示されている。制御論理回路を含むデータ・ユニット204に対するアドレスは、キャッシュ206への物理的アクセスを必要とする。キャッシュ206は、データがキャッシュに入っている場合、処理のためにキャッ

シュ206からフォーマッタ210にデータを渡す、この例では64ビットのデータ・ラインに接続された出力ポートを有する。

【0039】本発明の一実施形態では、命令がディスパッチされるたびにミスキュー・テーブル600内に項目が作成される。命令がデータ・キャッシュ内でヒットした場合、次のサイクルでその命令の項目がミスキュー・テーブル600から除去される。しかし、命令がデータ・キャッシュ内でミスした場合は、その実アドレスと、場合によっては有効アドレスと、その他の情報がミスキュー・テーブル600に残る。プロセッサはミスキュー・テーブル内の項目のアドレス情報を絶えず走査し、各サイクルで、プロセッサはミスキュー・テーブル600に記憶されている有効アドレスにあるキャッシュにアクセスを試みる。最終的に、ミスキュー・テーブル600内の各項目についてキャッシュ内でデータが入手可能になり、処理のためにフォーマッタに渡される。

【0040】上述のように有効アドレスを介してキャッシュにアクセスする代わりに、マイクロプロセッサはキャッシュに記憶されている実アドレスを介してキャッシュにアクセスを試みることもできることに留意されたい。これは、実施上の問題であり、本発明全体には影響しないことに留意されたい。

【0041】しかし、現行命令を最初からミスキュー・テーブル600に入れなくても本発明は機能することに留意されたい。現行命令は、理由を問わず、シーケンシング・ユニットによって最初に導入された直後に完了するものとして設定することができない場合にのみ、ミスキュー・テーブル600で示す必要がある。

【0042】好ましい実施形態では、命令項目はミスキュー・テーブル600にイン・オーダーで記憶される。これを図5に示す。命令項目410を生成する命令は命令項目420を生成する命令よりも古く、したがって命令項目410はミスキュー・テーブル600内で命令420よりも上位に記憶される。しかし、適切な識別情報と順序づけ情報があれば、ミスキュー・テーブルに命令を順不同で記憶することもできることに留意されたい。

【0043】ミスキュー・テーブル内の項目を図6に示す。本発明に必要な最低限の情報は、アドレス情報フィールド510と、アウト・オブ・オーダー情報フィールド520と、変更データ情報フィールド530である。アドレス情報は、実アドレス情報540と有効アドレス情報550を含むサブフィールドを有することもできる。ミスキュー・テーブル項目は、有効フィールド560や命令IDフィールド570など、それに付随するその他の情報を有することもでき、それらは順序づけ情報として使用できる。

【0044】他の実施形態では、ミスキュー・テーブル内の項目はアウト・オブ・オーダーでミスキュー・テーブルに記憶される。図6の有効フィールド560は、ミ

スキュー・テーブル項目が実際にまだミスキュー・テーブル600に入っているかどうかを示す。新しい項目は、有効フィールド560が設定されていない最初のミスキュー・テーブル行に作成される。命令IDフィールド570を使用して順序づけ情報が保持される。

【0045】次に図7を参照すると、プロセッサによって最初に、アドレスDにあるデータにアクセスする新規の命令800が渡される。好ましい実施形態では、ミスキュー・テーブル600内の次に使用可能なスロットに新規命令800の項目610が作成される。新規命令800はただちに完了し、プロセッサはミスキュー・テーブル600内のより古い項目620、630、及び640を新規命令800と照合する。具体的には、新規命令800のアドレスを、ミスキュー・テーブル600内のすべてのより古い項目のアドレス情報510と照合して一致するアドレスがないか調べる。当然ながら、ミスキュー・テーブル600内に含まれるすべての項目は新規命令800より古い命令に対応していなければならない。

【0046】新規命令800は次のサイクルで完了し、項目610が削除されることに留意されたい。また、項目610はただちに作成される必要はないことにも留意されたい。新規命令800のオペランド・アドレスをミスキュー・テーブル600の項目内のアドレス情報と照合し、新規命令800が次のクロック・サイクルで完了しない場合にのみ、その命令の項目610が作成されることになる。

【0047】代替実施形態では、項目610は削除されず、有効ミスキュー・テーブル項目ではなくなったことを示すように有効フィールドを設定する。その場合、ミスキュー・テーブル600内のその行を、新しい項目が使用することができる。

【0048】新規命令800のデータがキャッシュに入っているものとする。次に、新規命令800がキャッシュ内で見つかったデータを使用して完了するものと設定される。新規命令800が使用すると思われるアドレスを、ミスキュー・テーブル600内のすべてのより古い項目内のアドレス情報510と比較する。新規命令800のアドレスと一致するものが、ミスキュー・テーブル600内のより古い命令に対応する項目内のアドレス情報フィールド510と一致する場合、それらの一致項目はアウト・オブ・オーダーとしてマークされる。アウト・オブ・オーダーは、より新しい命令が、ミスキュー・テーブルにまだ入っているより古い命令も使用するアドレスにあるデータを使用して完了するとマークされていることを意味する。アウト・オブ・オーダー・フィールド520は項目をアウト・オブ・オーダーとしてマークするのに使用する。

【0049】図8に、次のクロック・サイクルで命令800が完了した後のミスキュー・テーブル600を示

す。命令項目620は、完了しておらず、新規命令800と同じ記憶場所にあるデータにアクセスするため、アウト・オブ・オーダー・フィールド520を使用するアウト・オブ・オーダーとしてマークされていることに留意されたい。

【0050】新規命令800がキャッシュからのデータを待ち続けなければならない場合、図9に示すようにミスキュー・テーブルに項目610として入れられ、完了可能になるまでそこにとどまる。図9には、その間に、ミスキュー・テーブル600に他のいくつかの命令項目が追加されている様子も図示されている。後の時点で、データ・キャッシュに、新規項目610には使用可能になっているが一致項目620には使用可能になっていないデータが入っている。したがって、新規命令800が完了した。

【0051】新規命令800が完了すると、プロセッサは命令800より古い命令に対応するミスキュー・テーブル内の項目、すなわち対応する項目610より上のすべての項目を走査する。プロセッサは、より古い項目を探索して新規命令項目のアドレス・フィールド630と一致するアドレスがないか調べる。新規命令項目610と同じアドレスに行く、より古い命令項目が見つかった場合、図10に示すようにその古い命令項目がアウト・オブ・オーダーとしてマークされる。

【0052】新規命令800が完了すると、ミスキュー600内のその項目が削除される。好ましい実施形態では、すべての他の項目が上に移動し、したがって図9及び図10に示すようにテーブルのタイミング順序が保持される。

【0053】任意の時点で命令のデータが変わった場合、その項目はそのことを示さなければならない。したがって、ミスキュー・テーブル600内に命令項目がある間にスヌープまたはその他のデータ変更の表示が行われると、対応する命令の変更標識530が潜在的な問題を示すように設定される。プロセッサは項目を走査し、データ変更時に同じアドレスを示すアドレスを有する各項目を、変更されているものとしてマークする。

【0054】データ・コヒーレンシが侵害された可能性がある場合、アウト・オブ・オーダー標識と変更標識の両方によって、それらの事象が発生したことが示される。すなわち、より古い命令より先に完了するものとして設定されていたより新しい命令に、より古い命令より古いデータが関連づけられている可能性がある。これによっておそらくデータ・コヒーレンシが侵害されることになる。

【0055】図11、図12、図13、及び図14に、この問題を検出するシーケンスを示す。ミスキュー・テーブル600は、それぞれ命令1210、1220、及び1230に対応する初期項目1110、1120、及び1130を有する。記憶場所aに行く新規命令124

0が発行され、そのデータがキャッシュ内で見つかる。すると、図12で項目1110はそれより前に新規命令1240が完了すると、アウト・オブ・オーダーとしてマークされる。図13で、記憶場所aが変更されたことを示すスヌープ要求1140が行われる。ミスキュー・テーブル600に問い合わせると、項目1110が変更済みとしてマークされている。したがって、項目1110はaから来るデータについてデータ・コヒーレンシ問題を示す。

【0056】新規命令1240は、ただちにクリアされる場合でもミスキュー・テーブル600に項目1150を加えたことに留意されたい。新規命令1240がキャッシュでミスした場合、加えられた項目1150はミスキュー・テーブル600内に残ることに留意されたい。また、項目1150はミスの後に作成することもできることにも留意されたい。重要なことは、新規命令1240のオペランド・アドレスがミスキュー・テーブル内の項目と比較され、新規命令1240がより古い項目1110、1120、及び1130に対応する命令よりも前に完了する場合に、一致するものがあればそれをマークする必要があることである。

【0057】項目1110の場合のように、命令項目が変更され、アウト・オブ・オーダーであることが示された場合、より古い命令項目にアウト・オブ・オーダーのマークを付けさせるそのより新しい命令を実行することはできない。好ましい実施形態では、項目1110に対応する命令が完了すると、ミスキュー・テーブルはその項目に対応する命令に問題が発生したことを完了論理回路に報告する。項目1110に対応する命令は完了し、実行することができる。しかし、その開始項目1110の後に続くすべての命令、すなわち命令1220、1230、及び1240と、完了して項目1110に対応する命令の後に実行するように設定された他の命令は取り消され、実行プロセス全体を再び通過するようにリセットされる。

【0058】本発明の好ましい実施形態では、アドレスはセット細分性とのみ比較される。したがって、アドレスはダブル・ワード境界のみと比較されて、潜在的な問題を示す。しかし、実アドレス及びその他の細分性でも比較可能であることに留意されたい。

【0059】本発明の好ましい実施形態では、エラー状況が検出された後、問題を指示命令の後のすべての命令がフラッシュされ、実行のためにリセットされる。したがって、残りの命令のフラッシュでデータ・コヒーレンシが保持される。

【0060】まとめとして、本発明の構成に関して以下の事項を開示する。

【0061】(1) 命令のアウト・オブ・オーダー実行のための装置であって、

a) 各項目がコンピュータ・マイクロプロセッサ内の命

令に対応する命令データの項目を保持するミスキュー・テーブルを含み、前記命令データは、

i) 命令のアドレス情報と、

ii) 他の命令を基準にした前記対応する命令の順序を示す命令の順序情報と、

iii) アドレス情報フィールドに対応するアドレスにあるデータを使用するより新しい命令が現行項目より前に完了したことを示すアウト・オブ・オーダー情報と、

iv) 前記アドレス情報フィールドに対応する前記アドレスにあるデータが変更された可能性を示す、命令のデータ変更情報とを含み、前記装置は、

b) 比較された項目内のアドレス情報フィールドに対応するアドレスにあるデータを使用する、より新しい命令が前記比較された項目より前に完了した場合、前記比較された項目のアウト・オブ・オーダー情報フィールドが設定される、前記ミスキュー・テーブル内の項目に前記アウト・オブ・オーダー情報フィールドを設定するアウト・オブ・オーダー比較機構と、

c) 前記ミスキュー・テーブル内の前記アドレス情報フィールド内のアドレス情報を、変更された可能性のあるアドレスと比較する変更比較機構とを含み、前記変更されたアドレスが比較された前記命令項目内の前記アドレス情報と一致する場合、前記項目内の前記変更フィールドが、前記アドレスにある変更されたデータを示すようにマークされる装置。

(2) アウト・オブ・オーダー命令と変更されたデータとを示す命令項目に対応する命令のすべての後続命令を取り消す、上記(1)に記載の装置。

(3) 前記アウト・オブ・オーダー比較機構が項目内のすべてのデータ・アドレス情報を比較して前記アドレスが一致するかどうかを判断する、上記(1)に記載の装置。

(4) 前記アウト・オブ・オーダー比較機構が、項目内の前記データ・アドレス情報の一部を比較して、前記アドレスが一致するかどうかを判断する、上記(1)に記載の装置。

(5) データ・コヒーレンシ侵害を生じさせる可能性のあるマイクロプロセッサ内のアウト・オブ・オーダー命令を検出する方法であって、

a) データ・アドレスを有する前記マイクロプロセッサ上で実行する新規命令を作成するステップと、

b) 前記新規命令が実行されるように設定されている場合、

i) 前記命令のデータ・アドレスをミスキュー・テーブル内の、前の命令に対応し、アドレス情報と命令順序情報とアウト・オブ・オーダー標識と変更データ標識とを含む既存の命令項目と比較し、

ii) 前記新規命令のデータ・アドレスがミスキュー・テーブル内の項目のアドレス情報と一致する場合、前記ミスキュー・テーブル内の前記一致項目をアウト・オブ・

10

20

30

40

50

オーダー命令にするステップと、

c) 前記新規命令が実行されるように設定されていない場合、前記ミスキュー・テーブル内の前記新規命令の項目を作成し、前記新規命令のデータ・アドレスが前記新規命令項目のアドレス情報に入れられ、前記命令項目の順序情報に前記命令の順序に関する情報が入れられる方法。

(6) d) 前記ミスキュー・テーブルを絶えず走査して、実行するように設定されている命令に対応する項目がないか調べるステップと、

e) 前記ミスキュー・テーブル内の項目に対応する命令が実行されるように設定されている場合、

i) 実行するように設定されている前記命令に対応する前記項目のアドレス情報を、実行するように設定されている前記命令より古い命令に対応する前記ミスキュー・テーブル内の項目のアドレス情報と比較するステップと、

ii) 前記ミスキュー・テーブル内に、実行するように設定されている前記命令より古い命令に対応する項目が見つかった場合、及び実行するように設定されている前記命令に対応する前記項目のアドレス情報が前記より古い命令に対応する前記項目内のアドレス情報と一致する場合、前記より古い命令に対応する前記項目をアウト・オブ・オーダーとしてマークするステップと、

iii) 実行するように設定されている前記命令に対応する前記項目を前記ミスキュー・テーブルから除去するステップとをさらに含む、上記(5)に記載の方法。

(7) f) データ変更事象がないか絶えず走査するステップと、

g) データ変更事象が発生した場合、

i) 前記変更されたデータのアドレスを前記ミスキュー・テーブルにブロードキャストするステップと、

ii) 前記変更されたデータのアドレスを前記ミスキュー・テーブルの項目内のアドレス情報と比較するステップと、

iii) 前記変更されたデータのアドレスと前記ミスキュー・テーブル内の項目のアドレスが一致する場合、前記一致するアドレスを有する前記ミスキュー・テーブル内の項目を変更されたものとしてマークするステップとをさらに含む、上記(6)に記載の方法。

【図面の簡単な説明】

【図1】 キャッシュ・バンクの1つにある同じデータにアクセスを試みる2つの命令を示す、インタリーブ・キャッシュの図である。

【図2】 図1のキャッシュ内のより古い命令の前にデータにアクセスする、より新しい命令と、アウト・オブ・オーダー完了がどのように行われるかを示す図である。

【図3】 スーパースカラ・プロセッサを示すブロック図

である。

【図4】 スーパースカラ・プロセッサ内のロード回路を示すブロック図である。

【図5】 本発明の好ましい実施形態によるミスキュー・テーブルを示す図である。

【図6】 各フィールドを示すミスキュー・テーブル内の項目の図である。

【図7】 本発明の一実施形態によってアウト・オブ・オーダー命令完了を検出する様子を示す図である。

10 【図8】 本発明の一実施形態によってアウト・オブ・オーダー命令完了を検出する様子を示す図である。

【図9】 本発明の一実施形態によってアウト・オブ・オーダー命令完了を検出する様子を示す図である。

【図10】 本発明の一実施形態によってアウト・オブ・オーダー命令完了を検出する様子を示す図である。

【図11】 本発明の一実施形態がどのように機能するかを示す図である。

【図12】 本発明の一実施形態がどのように機能するかを示す図である。

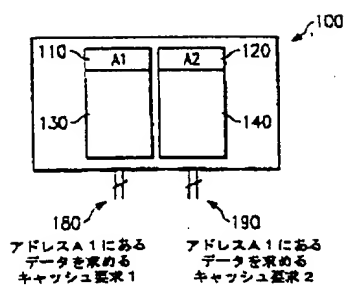
20 【図13】 本発明の一実施形態がどのように機能するかを示す図である。

【図14】 本発明の一実施形態がどのように機能するかを示す図である。

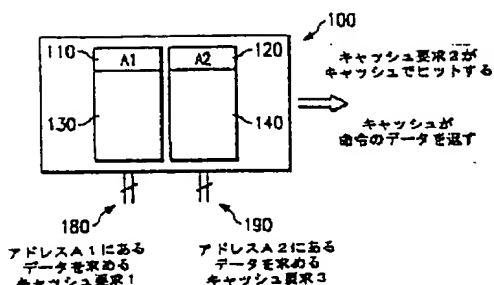
【符号の説明】

- 10 プロセッサ・システム
- 11 システム・バス
- 12 バス・インタフェース・ユニット
- 14 命令キャッシュ
- 16 データ・キャッシュ
- 30 18 シーケンサ・ユニット
- 20 分岐ユニット
- 22 固定小数点ユニットA
- 24 固定小数点ユニットB
- 26 複素固定小数点ユニット
- 28 ロード/ストア・ユニット
- 30 浮動小数点ユニット
- 32 汎用アーキテクチャ・レジスタ
- 34 固定小数点リネーム・バッファ
- 36 浮動小数点アーキテクチャ・レジスタ
- 40 38 浮動小数点リネーム・バッファ
- 39 システム・メモリ
- 40 特殊目的レジスタ
- 46 ディスパッチ・ユニット
- 202 加算器
- 204 データ・ユニット
- 206 キャッシュ
- 208 不整列/ビジー・ラッチ
- 210 フォーマッタ

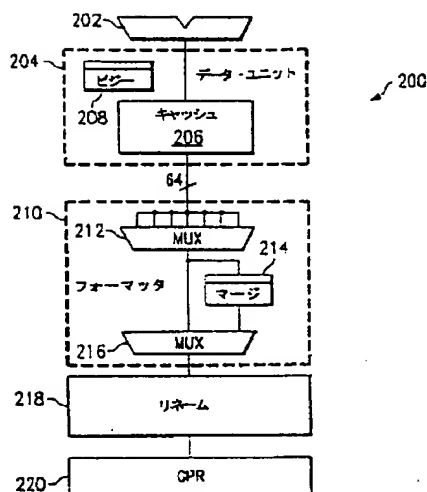
【圖 1】



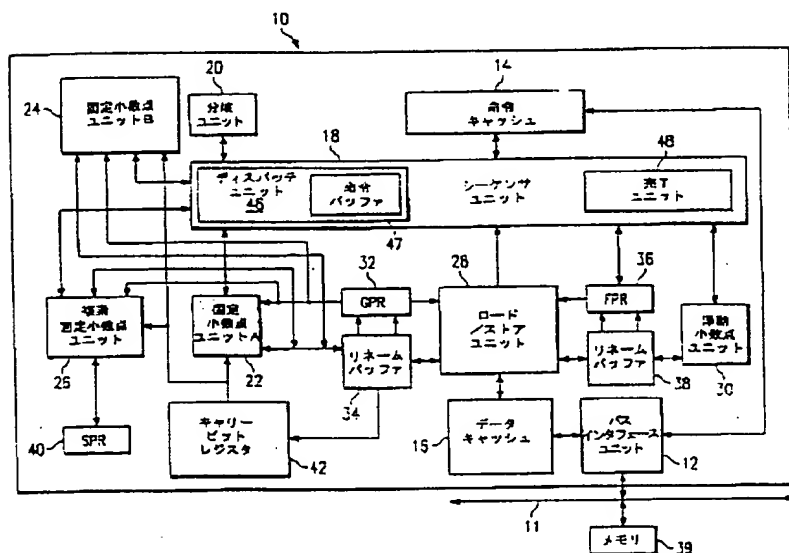
【圖 2】



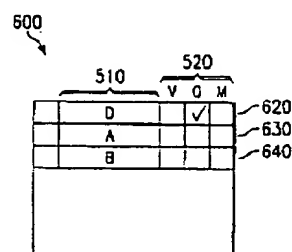
【圖 4】



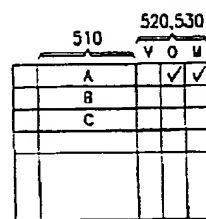
【图 3】



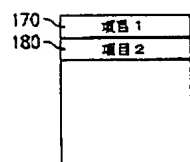
【图8】



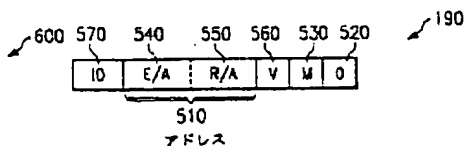
【图 14】



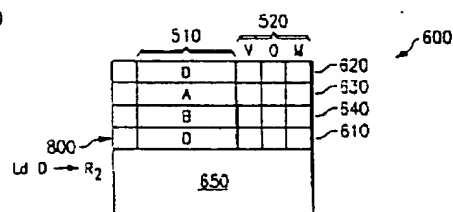
【图 5】



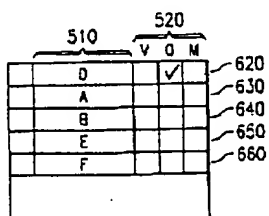
【図 6】



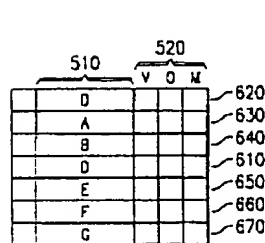
【图 7】



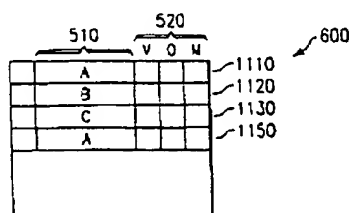
【圖 10】



【图9】



【图 11】



【図 1 2】

600

| 520 | | | | |
|-------|--|---|--|------|
| V O M | | | | |
| A | | ✓ | | 1110 |
| B | | | | 1120 |
| C | | | | 1130 |

【図 1 3】

| 510 | | 520 | | スレープ A |
|-------|--|-------|---|--------|
| V O M | | V O M | | |
| A | | | ✓ | 1140 |
| B | | | | |
| C | | | | |

フロントページの続き

(72) 発明者 デーヴィッド・スコット・レイ
 アメリカ合衆国 78628 テキサス州ジョー
 ジタウン ヤング・ランチ・ロード 700

(72) 発明者 ケビン・アーサー・キャロット
 アメリカ合衆国 78660 テキサス州フルー
 ガービル ヘイワース・コーブ 17907
 (72) 発明者 バリー・デュアン・ウィリアムソン
 アメリカ合衆国 78681 テキサス州ラウン
 ドロック リパーローン・ドライブ 807